```
*GIVE THE BANANAS TO THE MONKEYS ALTHOUGH THEY ARE NOT RIPE
*BECAUSE THEY ARE VERY HUNGRY!

*****

*****

TIME: 6 CPU SECS. 35 REAL SECS

121 STATE STACK CELLS USED
0 TOKEN STACK CELLS USED

155 DECISION POINTS
204 FAILURES
4047 CONSES
0 SECS GC TIME

DONNEZ LES BANANES AUX SINGES BIEN QU' ELLES NE SOIENT PAS MURES PARC~
E QU' ILS ONT TRES FAIM !

((((EX GIVE THE BANANAS) ((EX NIL NIL ((IMPCL)))) 4 (((THIS DTHIS) DUM~
THING) ((PRES ((*ENT OBJE) GIVE)) GIVE (DONNER)) ((THE (MUCH (((*ANI ~
SUBJ) ((TASTE SENSE) WANT)) (OBJE PLANT)))) BANANAS (FEMI BANANE)) NI~
L NIL NIL)) ((PTO THE MONKEYS) ((PTO (GIVE) RECI ((&PREOB A)))) 6 (((~
THIS DTHIS) DUMTHING) ((THIS PDO) PTO NIL) ((THE (MUCH ((MAN LIKE) BE~
AST))) MONKEYS (MASC SINGE)) NIL NIL NIL)) ((ALTHOUGH THEY ARE NOT RI~
PE) ((ALTHOUGH (GIVE) CONC (BIEN QUE (SUBCL)))) 1 (((THE (MUCH (((*AN~
I SUBJ) ((TASTE SENSE) WANT)) (OBJE PLANT)))) (THEY BANANAS) ((PRON 6~
 MASC PLUR))) ((NPRES (BE BE)) ARE ((IS_OBJECT HUNGRY) AVOIR (DIROB Q~
 FAIM)) ((IS_OBJECT THIRSTY) AVOIR (DIROB Q SOIF)) ((IS_OBJECT AFRAID~
) AVOIR (DIROB Q PEUR)) (ETRE)) (((PLANT POSS) ((*ANI (CAN USE)) KIND~
)) RIPE (MUR)) NIL NIL NIL)) ((BECAUSE THEY ARE VERY HUNGRY !) ((BECA~
USE (ARE) SOUR (PARCE QUE (INDCL)))) 1 (((THE (MUCH ((MAN LIKE) BEAST~
))) (THEY MONKEYS) ((PRON 6 MASC PLUR))) ((PRES (BE BE)) ARE ((IS_OBJ~
ECT HUNGRY) AVOIR (DIROB Q FAIM)) ((IS_OBJECT THIRSTY) AVOIR (DIROB Q~
 SOIF)) ((IS_OBJECT AFRAID) AVOIR (DIROB Q PEUR)) (ETRE)) (((*ANI POS~
S) ((((TASTE SENSE) WANT) STATE) KIND)) HUNGRY (AFFAME)) NIL (((MUCH ~
HOW) VERY (TRES))) NIL)))
*↑C

DO YOU WANT TO SEE THE DICTIONARY ? TYPE Y OR N FOLLOWED BY "RETURN"
*↑C
.R PPSAV
```

*THE COMPUTER'S PROGRAM MUST BE SENSITIVE TO GUIDING PRINCIPLES AND TO
*DISTURBING ANOMALIES. IT MUST GRASP LINGUISTIC REGULARITIES AND ALSO
*WAYWARDNESS. IDIOMS ARE METAPHORIC WRENCHES IN THE MACHINERY.

*****

*****

TIME: 6 CPU SECS, 35 REAL SECS

96 STATE STACK CELLS USED
0 TOKEN STACK CELLS USED

320 DECISION POINTS
414 FAILURES
5710 CONSES
2 SECS GC TIME

LE PROGRAMME DU CALCULATEUR DOIT ETRE SENSIBLE AUX PRINCIPES DIRECTEURS ET AUX ANOMALIES PERTURBATRICES . IL DOIT COMPRENDRE LES REGULARITES LINGUISTIQUES ET AUSSI LE COMPORTEMENT CAPRICIEUX . LES IDIOTISMES SONT DES CLEFS METAPHORIQUES DANS LE MECANISME .

```
(((THE COMPUTER'S PROGRAM MUST BE SENSITIVE) ((NIL NIL NIL
((INDCL)))) 1 (((THE ((COUNT GOAL) ((((MAN USE) (THING OBJE)) IN)
SIGN))) PROGRAM (MASC PROGRAMME)) ((MUPR (BE BE)) BE ((IS_OBJECT
HUNGRY) AVOIR (DIROB Q FAIM)) ((IS_OBJECT THIRSTY) AVOIR (DIROB Q
SOIF)) ((IS_OBJECT AFRAID) AVOIR (DIROB Q PEUR)) (ETRE)) (((*HUM
POSS) (((STATE OBJE) (MUCH FEEL)) KIND)) SENSITIVE (SENSIBLE))
(((((((COUNT SIGN) OBJE) USE) (THING SUBJ)) KIND) COMPUTER'S (DU
CALCULATEUR))) NIL NIL)) ((PTO GUIDING PRINCIPLES) ((PTO (SENSITIVE)
OBJE ((&PREOB A)))) 6 (((THIS DTHIS) DUMTHING) ((THIS PDO) PTO NIL)
((MUCH ((MUST GRAIN) SIGN)) PRINCIPLES (MASC PRINCIPE)) NIL NIL
(((WELL KIND) GUIDING (DIRECTEUR))))) ((AND PTO DISTURBING
ANOMALIES /.) ((AND NIL NIL (ET)) (PTO (SENSITIVE) OBJE ((&PREOB
A)))) 6 (((THIS DTHIS) DUMTHING) ((THIS PDO) PTO NIL) ((MUCH
(NOTPAIR SIGN)) ANOMALIES (FEMI ANOMALIE)) NIL NIL (((((MAN (
NOTPLEASE FEEL)) CAUSE) KIND) DISTURBING (PERTUBATEUR))))) ((IT MUST
GRASP LINGUISTIC REGULARITIES) ((NIL NIL NIL ((INDCL)))) 1 (((THE
((COUNT GOAL) (((MAN USE) (THING OBJE)) IN) SIGN))) (IT PROGRAM)
((IT_PRON))) ((MUPR ((*HUM SUBJ) ((SIGN OBJE) (TRUE THINK)))) GRASP
(COMPRENDRE)) ((MUCH (PAIR SIGN)) REGULARITIES (FEMI REGULARITE)) N
IL NIL ((((((MAN USE) (SIGN OBJE)) POSS) KIND) LINGUISTIC
(LINGUISTIQUE))))) ((AND ALSO WAYWARDNESS /.) ((AND NIL NIL (ET))
(NIL (GRASP) OBJE)) 3 (((NOTGRAIN STATE) WAYWARDNESS (MASC
COMPORTEMENT CAPRICIEUX)) ((THIS DBE) DUM DO) ((THIS DTHIS)
DUMTHING) NIL (((THIS HOW) ALSO (AUSSI))) NIL)) ((IDIOMS ARE
METAPHORIC WRENCHES) ((NIL NIL NIL ((INDCL)))) 1 (((MUCH (NOTGRAIN
SIGN)) IDIOMS (MASC IDIOTISME)) ((PRES (BE BE)) ARE ((IS_OBJECT
HUNGRY) AVOIR (DIROB Q FAIM)) ((IS_OBJECT THIRSTY) AVOIR (DIROB Q
SOIF)) ((IS_OBJECT AFRAID) AVOIR (DIROB Q PEUR)) (ETRE)) ((MUCH
((((THING OBJE) FORCE) GOAL) ((MAN USE) (OBJE THING)))) WRENCHES
(FEMI CLEF)) NIL NIL (((((((THING OBJE) PAIR) (SIGN SUBJ)) POSS)
KIND) METAPHORIC (METAPHORIQUE))))) ((IN THE MACHINERY /.) ((IN (ARE)
LOCA ((&PREOB DANS)) )) 6 (((THIS DTHIS) DUMTHING) ((THIS PDO) IN
NIL) ((THE (((((NOTSAME THING) OBJE) MAKE) GOAL) (MAN USE)) (OBJE
THING))) MACHINERY (MASC MECANISME)) NIL NIL NIL)))
```

Wilks: A small machine translation system based on deep semantic structures.

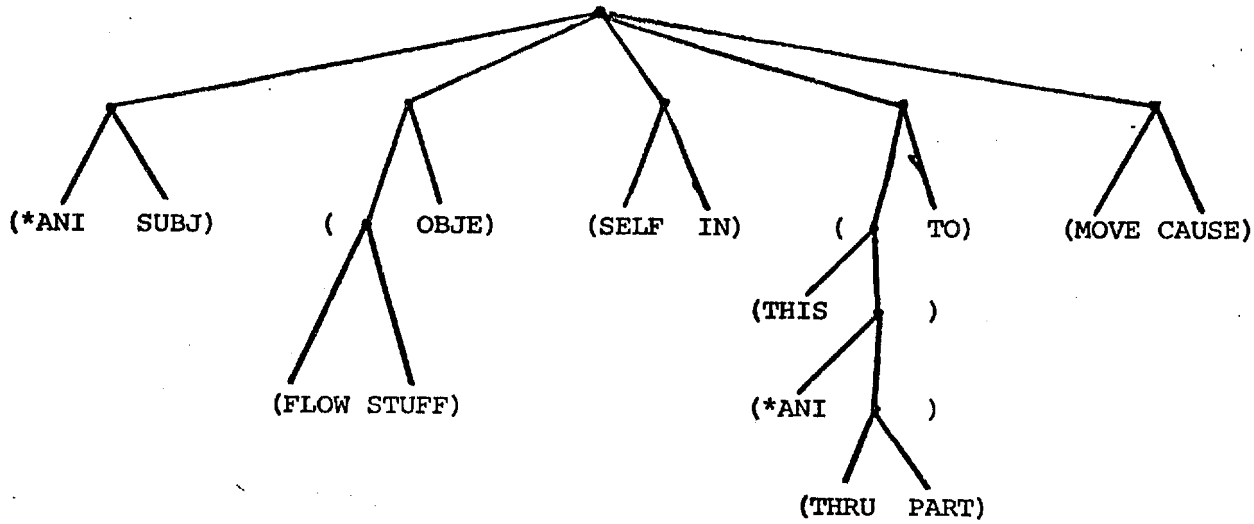
My own system constructs a semantic representation for small natural language texts: the basic representation is applied directly to the text and can then be 'massaged' by various forms of inference to become as deep as is necessary for well defined tasks demonstrating understanding. It is a uniform representation, in that information that might convenionally be considered as syntactic, semantic, factual or inferential is well expressed within a single type of structure. The fundamental unit in the construction of this meaning representation is the template, which is intended to correspond to an intuitive notion of a basic message of agent-action-object form. Templates are rigid format networks of more basic building blocks called formulas, which correspond to senses of individual words. In order to construct a complete text representation templates are bound together by two kinds of higher level structures called paraplates and inference rules. The templates themselves are built up as the construction of the representation proceeds, but the formulas, paraplates and inference rules are all present in the system at the outset and each of these three types of pre-stored structure is ultimately constructed from an inventory of eighty semantic primitive elements, and from functions and predicates ranging over those elements.

The system runs on-line as a package of LISP, MLISP and MLISP2 programs, taking as input small paragraphs of English, that can be made up by the user from a vocabulary of about 600 word senses, and producing a good French translation as output. This environment provides a pretty clear test of language understanding, because French translations for everyday prose are either right or wrong, and can be seen to be so, while at the same time, the major difficulties of understanding programs - word sense ambiguity, case ambiguity, difficult pronoun reference, etc. - can all be represented within a machine translation environment by, for example, choosing the words of the input sentence containing a pronoun reference difficulty so that the possible alternative references have different genders in French. In that way the French output makes quite clear whether or not the program has made the correct inferences in order to understand what it is translating. The program is reasonably robust in actual performance, and will even tolerate a certain amount of bad grammar in the input, since it does not perform a syntax analysis in the conventional sense, but seeks message forms representable in the semantic structures employed

Typical input would be a sentence such as 'John lives out of town and drinks his wine out of a bottle. He then throws the bottles out of the window.' The program will produce French sentences with different output for each of the three occurrences of 'out of', since it realises that they function quite differently on the three occasions of use, and that the difference must be reflected in the French. A sentence such as 'Give the monkeys bananas although they are not ripe because they are very hungry' produces a translation with different equivalents for the two occurrences of 'they', because the system correctly realises, from what I shall describe below as preference considerations, that the most sensible interpretation is one in which the first 'they' refers to the bananas and the second to the monkeys, and bananas and monkeys have different genders in French. These two examples are dealt with in the 'basic mode' of the system. (Wilks 73a) In many cases it cannot resolve pronoun ambiguities by the sort of straightforward 'preference considerations' used in the last example, where, roughly speaking, 'ripeness' prefers to be predicated of plant-like things, and hunger of animate things. Even in a sentence as simple as 'John drank the wine on the table and it was good', such considerations are inadequate to resolve the ambiguity of 'it' between wine and table, since both may be good things. In such cases, of inability to resolve within its basic mode, the program deepens the representation of the text so as to try and set up chains of inference that will reach, and so prefer, only one of the possible referents. I will return to these processes in a moment, but first I shall give some brief description of the basic representation set up for English.

For each sense of a word in its dictionary the program sees a formula. This is a tree structure of semantic primitives, and is to be interpreted formally using dependency relations. The main element in any formula is the rightmost, called its head, and that is the fundamental category to which the formula belongs. In the formulas for actions, for example, the head will always be one of the primitives PICK, CAUSE, CHANGE, FEEL, HAVE, PLEASE, PAIR, SENSE, USE, WANT, TELL, BE, DO, FORCE, MOVE, WRAP, THINK, FLOW, MAKE, DROP, STRIK, FUNC or HAPN.

Here is the tree structure for the action of drinking:



Once again, it is not necessary to explain the formalism in any detail, to see that this sense of 'drink' is being expressed as a causing to move a liquid object (FLOW STUFF) by an animate agent, into that same agent (containment case indicated by IN, and formula syntax identifies SELF with the agent) and via (direction case) an aperture (THRU PART) of the agent.

Template structures, which actually represent sentences and their parts are built up as networks of formulas like the one above. Templates always consist of an agent node, and action node and an object node, and other nodes that may depend on these. So, in building a template for 'John drinks wine', the <u>whole of</u> the above tree-formula for 'drinks' would be placed at the central action node, another tree structure for 'John' at the agent node and so on. The complexity of the system comes from the way in which the formulas, considered as active entities, dictate how other places in the same template should be filled.

Thus, the 'drink' formula above can be thought of as an entity that fits at a template action node, and seeks a liquid object, that is to say a formula with (FLOW STUFF) as its right-most branch, to put at the object node of the same template. This seeking is preferential, in that formulas not satisfying that requirement will be accepted, but only if nothing satisfactory can be found. The template finally established for a fragment of text is the one in which the most formulas have their preferences satisfied. There is a general principle at work here, that the right interpretation 'says the least' in information-carrying terms. This very simple device is able to do much of the work of a syntax and word-sense ambiguity resolving program. For example, if the sentence had been 'John drank a whole pitcher', the formula for the 'pitcher of liquid' would have been preferred to that for the human, since the subformula (FLOW STUFF) could be appropriately located within it.

A considerable amount of squeezing of this simple canonical form of template is necessary to make it fit the complexity of language: texts have to be fragmented initially; then, in fragments which are, say, prepositional phrases there is a dummy agent imposed, and the prepositional phrases there is a dummy agent imposed, and the prepositional formula functions as a pseudo-action. There are special 'less preferred' orders to deal with fragments not in agent-action-object order, and so on.

When the local inferences have been done that set up the agent-action-object templates for fragments of input text, the system attempts to tie these templates together so as to provide an overall initial structure for the input. One form of this is the anaphora tie, of the sort discussed for the monkeys and bananas example above, but the more general form is the case tie. Assignment of these would result in the template for the last clause of 'He ran the mile in a paper bag' being tied to the action node of the template for the first clause ('He ran the mile'), and the tie being labelled CONTainment. These case ties are made with the aid of another class of ordered structures, essentially equivalent to Fillmore's case frames, called paraplates and which are attached to the formulas for English prepositions. So, for 'outof', for example, there would be at least six ordered paraplates, each of which is a string of functions that seek inside templates for information. In general, paraplates range across two, not necessarily contiguous, templates. So, in analysing 'He put the number he thought of in the table', the successfully matching paraplate would pin down the dependence of the template for the last of the three clauses as DIREction, by taking as argument only that particular template for the last clause that contained the formula for 'a numerical table', (and not a template representing a kitchen table) and it would do that because of a function in that paraplate seeking a similarity of head (SIGN in this case) between the two appropriate object formulas, for 'number' and 'table'. The other template containing the 'furniture' formula for 'table' would naturally not satisfy the function because SIGN would not be the head of this sense formula for 'table'.

The structure of mutually connected templates that has been put together thus far constitutes a 'semantic block', and, if it can be constructed, then as far as the system is concerned all semantic and referential ambiguity has been resolved and it will begin to generate French by unwrapping the block again. The generation aspects of this work have been described in (Herskovits '73). One aspect of the general notion of preference is that the system should never construct a deeper or more elaborate semantic representation than is necessary for the task in hand and, if the initial block can be constructed and a generation of French done, no 'deepening' of the representation will be attempted.

However, many examples cannot be resolved by the methods of this 'basic mode' and, in particular, if a word sense ambiguity, or pronoun reference, is still unresolved, then a unique semantic block of templates cannot be constructed and the 'extended mode' will be entered.* In this mode, new template-like forms are extracted from existing ones, and then added to the template pool from which further inferences can be made. So, in the template derived earlier for 'John drinks wine', the system enters the formula for 'drinks', and draws inferences corresponding to each case sub-formula. In this example it will derive template-like forms equivalent to, in ordinary English, 'The wine is in John', 'The wine entered John via an aperture' and so on. The extracted templates express information already implicitly present in the text, even though many of them are partial inferences: ones that may not necessarily be true.

Common-sense inference rules are then brought down, which attempt, by a simple strategy, to construct the shortest possible chain of rule-linked template forms from one containing an ambiguous pronoun, say, to one containing one of its possible referents. Such a chain then constitutes a solution to the ambiguity problem, and the preference approach assumes that the shortest chain is always the right one. So, in the case of 'John drank the wine /on the table/ and it was good', (in three template-matching fragments as shown) the correct chain to 'wine' uses the two rules

* Wilks '73b and (in press)

I 1. ((*ANI 1) ((SELF IN) (MOVE CAUSE)) (*REAL 2)) → (1(*JUDG) 2)

or, in 'semi-English',

[animate-1 cause-to-move-in-self real-object-2] → [1 *judges 2]

I 2. (1 BE (GOOD KIND)) ↔ ((*ANI 2) WANT 1)

or, again,

[1 is good] ↔ [animate-2 wants 1]

These rules are only partial, that is to say, they correspond only to what we may reasonably look out for in a given situation, not to what MUST happen. The hypothesis here is that understanding can only take place on the basis of simple rules that are confirmed by the context of application. In this example the chain constructed may be expressed as (using the above square bracket notation to contain not a representation, but simply an indication, in English, of the template contents):

forwards inf.	[John drank	the-wine]	Template 1
	[John causes-to-move-in-self	wine]	÷ Template 1
	[John * judges	wine]	by I 1.
backwards inf.	[John wants	wine]	÷ line above
	[wine is	good]	by I 2.
	[?it is	good]	Template 3

The assumption here is that no chain using other inference rules could have reached the 'table' solution by using less than two rules.

The chief drawback of this system is that codings consisting entirely of primitives have a considerable amount of both vagueness and redundancy. For example, no reasonable coding in terms of structured primitives could be expected to distinguish, say, 'hammer' and 'mallet'. That may not matter provided the codings can distinguish importantly different senses of words. Again, a template for the sentente 'The sheperd tended his flock' would contain considerable repetition, each node of the template trying, as it were, to tell the whole story by itself. Again, the preference criteria are not in any way weighted, which might seem a drawback, and the preferential chain-length criterion for inference chains might well seem too crude. Whether or not such a system can remain stable with a considerable vocabulary, of say several thousand words, has yet to be tested.

FIGURES

On the next sheet is a full template---a simple one for "John shut the door" consisting of only three formulas.

On the following sheets are (rather feint) xeroxes of system output---the first resolves two "they" s into different French pronouns and the second deals with a simple metaphor.

The basic references from the text of the handout are:

Wilks,Y.,An intelligent Analyzer and Understander of English, Communications of the A.C.M., 1975

and,on the generative aspects of the program,

Herskovits,A, The Generation of French from a Semantic Structure, Memo No. 212 Stanford Artificial Intelligence Lab.,1973.

Note:the large blocks of code at the bottom of the computer output sheets are "semantic blocks"(Q.V. in text): compressed forms of templates as on the next sheet,plus ties between such templates,plus the French generative grammar (i.e.French words,phrases,whole forms of verbs if irregular,and patterns dictating the French output are all inside this "block".

"John shut the door"